Name: Lutika Kolhe

Core Java Assignment 6

# Assignments on Collections

1. Given a Tree Map<Long, Contact> which has phone numbers for keys and contact objects for values.

Write solutions to

* Fetch all the keys and print them,
* Fetch all the values and print them
* Print all key-value pairs

Note:

* Contacts should be stored in descending order of phone number
* Contact Class:

PhoneNumer: Name: Email: Gender:

Ans- I created 2 Class here first is Tree.java and another is Contact.java

Class 1= Tree.java

**import** java.util.Collections;

**import** java.util.Map;

**import** java.util.Set;

**import** java.util.TreeMap;

**import** tree.contact.gender;

**public** **class** Tree1 {

**public** **static** **void** main(String[] args)

{

Map<Long,Contact> map=**new** TreeMap<Long,Contact>();

Contact c1=**new** Contact((**long**)89034567,"ram","ram@.com",gender.m);

Contact c2=**new** Contact((**long**)12345679,"raj","raj@.com",gender.m);

Contact c3=**new** Contact((**long**)67864747,"sam","sam@.com",gender.Fe);

Contact c4=**new** Contact((**long**)45754757,"tom","tom@.com",gender.m);

map.put((**long**)89034567, c1);

map.put((**long**)12345679, c2);

map.put((**long**)67864747, c3);

map.put((**long**)45754757, c4);

**for**(Map.Entry<Long, Contact> entry:map.entrySet()){

Long key=entry.getKey();

Contact c=entry.getValue();

System.***out***.println(key+" Details:");

System.***out***.println(c.phoneno+" "+c.name+" "+c.email+" "+c.g);

}

System.***out***.println("............");

System.***out***.println("After Sorted:");

Map<Long,Contact> sortedMapDesc = **new** TreeMap<>(

Collections.*reverseOrder*());

sortedMapDesc.putAll(map);

**for**(Map.Entry<Long,Contact> entry1 : sortedMapDesc.entrySet())

{

Long key=entry1.getKey();

Contact c8=entry1.getValue();

System.***out***.println(key+" Details:");

System.***out***.println(c8.phoneno+" "+c8.name+" "+c8.email+" "+c8.g);

}

}

}

Class 1= Contact.java

**package** tree;

**import** java.util.EnumSet;

**public** **class** Contact {

**long** phoneno;

String name,email;

**public** **enum** gender {***F***,***M***}

gender g;

/\*\*

\* **@param** phoneno

\* **@param** name

\* **@param** email

\* **@param** g

\*/

**public** Contact(**long** phoneno, String name, String email, gender g) {

**super**();

**this**.phoneno = phoneno;

**this**.name = name;

**this**.email = email;

**this**.g = g;

}

**public** **long** getPhoneno() {

**return** phoneno;

}

**public** **void** setPhoneno(**long** phoneno) {

**this**.phoneno = phoneno;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** gender getG() {

**return** g;

}

**public** **void** setG(gender g) {

**this**.g = g;

}

}

**Output:**

12345679 Details:

12345679 raj raj@.com m

45754757 Details:

45754757 tom tom@.com m

67864747 Details:

67864747 samsam@.com Fe

89034567 Details:

89034567 ram ram@.com m

............

After Sorted:

89034567 Details:

89034567 ram ram@.com m

67864747 Details:

67864747 samsam@.com Fe

45754757 Details:

45754757 tom tom@.com m

12345679 Details:

12345679 raj raj@.com m

1. Write an application to store 10 unique product objects. In case there is an attempt to add a duplicate product, it should be silently rejected.

Ans- I created 2 Class here first is AddDuplicateProduct.java and another is Hash.java

Class 1= AddDuplicateProduct.java

**public** **class** AddDuplicateProduct **implements** Comparable<AddDuplicateProduct>

{

**private** String product\_name;

**private** **int** product\_id;

AddDuplicateProduct(String product\_name, **int** product\_id)

{

**this**.product\_id = product\_id;

**this**.product\_name = product\_name;

}

**private** String getName()

{

**return** product\_name;

}

**public** **int** getId()

{

**return** product\_id;

}

**public** **int** compareTo(AddDuplicateProduct f)

{

**if**(product\_id == f.getId())

{

**return** 0;

}

**else** **if**(product\_name.compareTo(f.getName()) < 0)

{

**return** -1;

}

**else**

{

**return** -1;

}

}

**public** String toString()

{

**return** product\_name + " - " + product\_id;

}

}

Class 2= DublicateMain.java

**import** java.util.TreeSet;

**public** **class** DublicateMain {

**public** **static** **void** main(String[] args) {

TreeSet<AddDuplicateProduct> dublicate = **new** TreeSet<>();

dublicate.add(**new** AddDuplicateProduct("idea",1));

dublicate.add(**new** AddDuplicateProduct("airtail",2));

dublicate.add(**new** AddDuplicateProduct("jio",3));

dublicate.add(**new** AddDuplicateProduct("vodaphone",4));

dublicate.add(**new** AddDuplicateProduct("jio",2));

dublicate.add(**new** AddDuplicateProduct("idea",5));

**for**(AddDuplicateProduct d : dublicate)

{

System.***out***.println(d);

}

}

}

**Output:**

idea - 5

vodaphone - 4

jio - 3

airtail - 2

idea - 1

1. Store atleast 10 Employee objects in a TreeSet. When your application runs the user should be asked to selct one of the options upon which you will print the employee details in a sorted manner.

Ans- I created 6 Class here first is Employee.java, Id.jav, Name.java, Salary.java, Department.java, and Final.java

Class 1= Employee.java

**public** **class** Employee {

**private** **int** id;

**private** **int** salary;

**private** String name;

**private** String department;

**public** Employee (**int** id, **int** salary, String name, String department)

{

**this**.id = id;

**this**.salary = salary;

**this**.name = name;

**this**.department = department;

}

**public** String getName()

{

**return** name;

}

**public** **void** setName(String name)

{

**this**.name = name;

}

**public** String getDep()

{

**return** department;

}

**public** **void** setDep(String department)

{

**this**.department = department;

}

**public** **int** getId()

{

**return** id;

}

**public** **int** getSalary()

{

**return** salary;

}

}

Class 2= Id.java

**import** java.util.Comparator;

**public** **class** Id **implements** Comparator<Employee>

{

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getId() - o2.getId();

}

}

Class 3= Name.java

**import** java.util.Comparator;

**public** **class** Name **implements** Comparator<Employee>

{

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getName().compareTo(o2.getName());

}

}

Class 4= Salary.java

**import** java.util.Comparator;

**public** **class** Salary **implements** Comparator<Employee>{

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getSalary() - o2.getSalary();

}

}

Class 5= Department.java

**import** java.util.Comparator;

**public** **class** Department **implements** Comparator<Employee>

{

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getDep().compareTo(o2.getDep());

}

}

Class 6= Final.java

**import** java.util.\*;

**public** **class** Final {

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

System.***out***.println("You want to sort in order of \n\n1.ID\n2.Name\n3.Department\n4.Salary\n\nEnter your option: ");

**int** option = scan.nextInt();

**switch**(option)

{

**case** 1:

TreeSet<Employee> set = **new** TreeSet<Employee>(**new** Id());

set.add(**new** Employee(1,80000,"Lutika","A"));

set.add(**new** Employee(2,81000,"Niharika","B"));

set.add(**new** Employee(3,82000,"Harshita","C"));

System.***out***.println("Increasing Order with the Id : ");

**for**(Employee o : set)

{

System.***out***.print(o.getId()+","+o.getName()+","+o.getDep()+","+o.getSalary());

System.***out***.println();

}

**break**;

**case** 2:

TreeSet<Employee> setN = **new** TreeSet<Employee>(**new** Name());

setN.add(**new** Employee(1,80000,"Lutika","A"));

setN.add(**new** Employee(2,81000,"Niharika","B"));

setN.add(**new** Employee(3,82000,"Harshita","C"));

System.***out***.println("Increasing Order with the Name : ");

**for**(Employee o : setN)

{

System.***out***.print(o.getId()+","+o.getName()+","+o.getDep()+","+o.getSalary());

System.***out***.println();

}

**break**;

**case** 3:

TreeSet<Employee> setD = **new** TreeSet<Employee>(**new** Department());

setD.add(**new** Employee(1,80000,"Lutika","A"));

setD.add(**new** Employee(2,81000,"Niharika","B"));

setD.add(**new** Employee(3,82000,"Harshita","C"));

System.***out***.println("Increasing Order with the Department : ");

**for**(Employee o : setD)

{

System.***out***.print(o.getId()+","+o.getName()+","+o.getDep()+","+o.getSalary());

System.***out***.println();

}

**break**;

**case** 4:

TreeSet<Employee> setS = **new** TreeSet<Employee>(**new** Salary());

setS.add(**new** Employee(1,80000,"Lutika","A"));

setS.add(**new** Employee(2,81000,"Niharika","B"));

setS.add(**new** Employee(3,82000,"Harshita","C"));

System.***out***.println("Increasing Order with the Salary : ");

**for**(Employee o : setS)

{

System.***out***.print(o.getId()+","+o.getName()+","+o.getDep()+","+o.getSalary());

System.***out***.println();

}

**break**;

}

}

}

**Output:**

You want to sort in order of

1.ID

2.Name

3.Department

4.Salary

Enter your option:

3

Increasing Order with the Department :

1,Lutika,A,80000

2,Niharika,B,81000

3,Harshita,C,82000

1. Given a Linked List of Objects representing date of birth's (use any inbuild java class to represent date), print the date's along with the message: Your date of Birth is DD-MM-YYYY, and it (was or was not) a leap year.

* E.g. For the date 23-12-2000

Your date of birth is 23-12-2000 and it was a leap year

**import** java.time.LocalDate;

**import** java.time.format.DateTimeFormatter;

**import** java.util.LinkedList;

**public** **class** LeapYear {

**public** **static** **void** main(String[] args) {

LocalDate date1 = LocalDate.*of*(1997, 06, 03);

LocalDate date2 = LocalDate.*of*(2016, 11, 7);

LocalDate date3 = LocalDate.*of*(1999, 10, 11);

LinkedList<LocalDate> list = **new** LinkedList<LocalDate>();

list.add(date1);

list.add(date2);

list.add(date3);

**for**(LocalDate l : list)

{

String printDate = l.format(DateTimeFormatter.*ofPattern*("dd-MM-YYYY"));

**if**(l.isLeapYear())

{

System.***out***.println("Your Date of Birth is " + printDate + " and it was a leap year");

}

**else**

{

System.***out***.println("Your Date of Birth is " + printDate + " and it was not a leap year");

}

}

}

}

**Output:**

Your Date of Birth is 03-06-1997 and it was not a leap year

Your Date of Birth is 07-11-2016 and it was a leap year

Your Date of Birth is 11-10-1999 and it was not a leap year